Managing dot-files

BV

[2015-09-20 Sun 11:14]

1 The problem

Like most people nowadays, I have multiple accounts on multiple computers serving multiple uses. As new accounts are created, new computers purchased, new configuration created, it's challenging to keep it all synchronized. In the past I've tried a scheme of various and independent git repositories holding configuration files and ad-hoc methods to "hook" them into where they need to be found. This sort of works but the various clones are always diverging and new accounts take effort to set up.

2 The solution

Obviously, this is not a unique problem and good effort has been done to try and solve it. For my next iteration I am taking an approach based on this nice blog post. It includes these ingredients

vcsh to allow multiple git repos to "overlap" in `$HOME`

myrepos to efficiently keep these repos in sync

gitolite to give me private git hosting

and of course bash, emacs, ssh, git, etc.

It is only after working through this approach did I really begin to appreciate it.

3 First Time Setup

Initial bootstrapping of this is described here. Two computers are involved:

hal laptop

haiku server

A gitolite3 server is set up on `haiku` to hold various git repos, one per application. Then vcsh and myrepos are configured on the `hal`.

3.1 Prep SSH

Too many SSH keys means SSH connection failures if the correct key isn't used within the first ~3 tries. So, configure SSH on `hal` to explicitly use a key for accessing gitolite3's admin user. The host defined here will be used to access gitolite.

```
$ cat <EOF >> ~/.ssh/config
Host gitolite-haiku-admin
     User gitolite3
     Hostname haiku
     ForwardAgent no
     ForwardX11 no
     IdentityFile ~/.ssh/gitolite-server
Host gitolite-haiku
     User gitolite3
     Hostname haiku
     ForwardAgent no
     ForwardX11 no
     IdentityFile ~/.ssh/id_rsa
EOF
```

3.2 gitolite3

Install gitolite3 on my home `haiku` server, using `~/.ssh/gitolite-server.pub` as the admin key.

```
$ git clone gitolite-haiku-admin:gitolite-admin gitolite-admin-haiku
```

Add my user key some `dot-*` repos, starting with one to hold `.bbdb` as a simple example.

```
$ cp ~/.ssh/id_rsa.pub keydir/bv.pub
$ git add keydir/bv.pub
$ cat <EOF >> conf/gitolite.conf
```

```
repo dot-bbdb
    RW+ = bv
repo dot-myrepos
    RW+ = bv
EOF
$ git commit -a -m "Start up."
$ git push
```

3.3 `vcsh` and first dotfile

On `hal` install vcsh, initialize a repo for bbdb:

```
$ sudo apt-get install vcsh
$ vcsh init bbdb
$ vcsh bbdb add .bbdb
$ vcsh bbdb commit -m "First commit"
$ vcsh bbdb remote add origin gitolite-haiku:dot-bbdb
$ vcsh bbdb push -u origin master
$ cat <EOF >> ~/.gitignore.d/bbdb
*
!/.bbdb
EOF
$ vcsh bbdb status
On branch master
Your branch is up-to-date with 'origin/master'.
nothing to commit, working directory clean
```

3.4 `myrepo`

```
$ cd ~
$ vcsh clone git@github.com:RichiH/vcsh_mr_template.git mr
$ vcsh mr remote set-url origin gitolite-haiku:dot-myrepos
```

Also edit `~/.config/mr/available.d/mr.vcsh` to have this new repo. Likewise, add a `mr` config fragment for `bbdb`:

```
$ cat .config/mr/available.d/bbdb.vcsh
[$HOME/.config/vcsh/repo.d/bbdb.git]
checkout = vcsh clone gitolite-haiku:dot-bbdb bbdb
$ cd .config/mr/config.d/
$ ln -s ../available.d/bbdb.vcsh .
$ mr status
$ mr commit -m "..."
$ mr push
```

3.5 ssh

One more example. SSH is a whole config world unto itself. See this topic.

```
$ vcsh init ssh
$ vcsh ssh add .ssh/.gitignore ...
$ vcsh ssh commit -m '...'
$ vcsh ssh remote add origin gitolite-haiku:dot-ssh
$ vcsh ssh push -u origin master
```

Also tweak:

- `.config/mr/*.d/`
- `.gitignore.d/ssh`

4 Bootstrap new account

First one may need to add a new client's SSH key to gitolite:

```
$ cd gitolite-admin-haiku
$ cp /path/to/key.pub keydir/<client>/<user>.pub
$ emacs conf/gitolite.conf   #<-- only if <user> is new
$ git add keydir/<client>/<user>.pub
$ git commit ...
$ git push
```

Then, there is a bit of a catch-22 bootstrap issue as we need ssh config to access the dot files git repos properly but we can't get that until we get dot-ssh. This breaks the loop:

```
cat <EOF >> .ssh/config
Host gitolite-haiku
     User gitolite3
     Hostname haiku
     ForwardAgent no
     ForwardX11 no
     IdentityFile ~/.ssh/privkeys/id_gitolite-%u-%l-%r-%h
     PreferredAuthentications publickey
EOF
```

Then, "just" (this is a lie) do

```
$ vcsh clone gitolite3@haiku:dot-myrepos mr
$ mr up
```

Some errors may occur from the `install.sh` script in the `ssh` area.

```
$ mkdir ~/.ssh/config.d/haiku
$ mv ~/.ssh/config ~/.ssh/config.d/haiku/local.cfg
$ cd ~/.ssh/config.d/haiku
$ ln -s ../fragments/[...] .
$ ~/.ssh/install.sh
```

Other errors will occur due to `.bashrc` or other files getting in the way of `vcsh` controlled copies. Move them aside and thrash about to get things working smoothly.

One odd problem is that files under `~/.gitignore.d/` somehow get added to the individual repositories they control.